# Osnove HTTP-a

HTTP (HyperText Transfer Protocol) služi najpre za prenos HTML-a (HyperText Markup Language). Baziran je na zahtev–odgovor modelu, što znači da pre nego što nam server išta pošalje, mi kao klijent moramo poslati zahtev serveru. Zahtevi su standardizovani u formi metoda. Na osnovu radnje koju klijent želi da izvrši, zahtevi se mogu podeliti na one koji samo dobavljaju podatke sa servera i one koji menjaju podatke na serveru. HTTP zahtevi su:

- GET
- POST
- PUT
- HEAD
- DELETE
- PATCH
- OPTIONS

Nakon svakog zahteva upućenog ka serveru, server nam vraća odgovor koji između ostalog sadrži i statusni kod, koji bliže objašnjava šta se desilo sa tim zahtevom, a koji odgovara odgovarajućoj statusnoj poruci, o čemu je već bilo reči. Najčešći statusni kodovi su:

- 200 – OK
- 204 – No Content
- 304 – Not Modified
- 400 – Bad Request
- 403 – Forbidden
- 404 – Not Found
- 405 – Not Allowed
- 500 – Internal Server Error

## Python kao HTTP klijent

Iako su pregledači (browseri) najčešći vid HTTP klijenta, klijent može biti svaki program koji ima pristup mreži te može slati zahteve i primati odgovore od servera. U Pythonu, slanje zahteva se vrši pomoću već izgrađenih biblioteka. Najpoznatije biblioteke za slanje zahteva ka HTTP serveru su urllib.request i requests. Prva biblioteka je ugrađena, a druga se mora instalirati koristeći *pip install requests* komandu iz komandnog prozora. S tim u vezi, requests biblioteka je mnogo moćnija i lakša za upravljanje od jednostavnije urllib.request biblioteke. Primeri slanja zahteva pomoću Python koda će se zasnivati na slanju zahteva ka adresi http://httpbin.org/. Ovaj sajt, delo osobe koja je zaslužna i za kreiranje requests biblioteke, omogućava nam jednostavno testiranje HTTP biblioteka.

Dokumentacija za urllib.request se može naći na stranici koja je deo zvanične dokumentacije Pythona, dok se dokumentacija za requests biblioteku može naći na ovoj stranici.

Kroz primere ćemo pokazati kako se zahtevi i funkcionalnosti HTTP servera implementiraju u biblioteke urllib.request i requests.

Sajt na kojem testiramo ove biblioteke za svaku metodu HTTP-a pruža krajnju tačku (link) koja odgovara imenu metode koju testiramo. Za GET bi krajnja tačka izgledala ovako: http://httpbin.org/get, dok bi na primer za POST metodu izgledala ovako: http://httpbin.org/post.

## GET zahtev ka HTTP serveru

### Biblioteka urllib.request

Kako bismo poslali GET zahtev, potrebno je prvenstveno uvesti urllib.request biblioteku. Za kreiranje GET zahteva pomoću ove biblioteke koristimo metodu urlopen() kojoj prosleđujemo link stranice koju želimo da dobavimo.

```
urllib.request.urlopen('https://httpbin.org/get')
```

Rezultat metode urlopen() (povratna vrednost) jeste odgovor koji dobijamo od servera. Ako taj odgovor dodelimo nekoj promenljivoj, na primer po imenu response, možemo pristupiti detaljima tog odgovora. Povratna vrednost ove funkcije je tipa http.client.HTTPResponse, koji podržava atribute i metode od kojih su često korišćeni: url, code, info(), read(), kao i readlines().

**Primer slanja GET zahteva pomoću urllib.request biblioteke**

```
import urllib.request

response = urllib.request.urlopen('https://httpbin.org/get')

print("URL: ",response.url)

print("Code: ",response.code)

print("Info: ", response.info())

print("Read: ", response.read())

print("Readlines: ",response.readlines())
```

URL: https://httpbin.org/get

Code: 200

Info: Date: Mon, 29 Jun 2020 01:04:01 GMT

Content-Type: application/json

```
Content-Length: 275

Connection: close

Server: gunicorn/19.9.0

Access-Control-Allow-Origin: *

Access-Control-Allow-Credentials: true

Read: b'{\n  "args": {}, \n  "headers": {\n    "Accept-Encoding": "identity", \n    "Host":
"httpbin.org", \n    "User-Agent": "Python-urllib/3.7", \n    "X-Amzn-Trace-Id": "Root=1-
5ef93e01-65bee1260e0ae9a84c7c55ba"\n  }, \n  "origin": "24.135.243.74", \n  "url":
"https://httpbin.org/get"\n}\n'

Readlines:  []
```

Na ovom ispisu uviđamo čemu koji poziv služi:

- url – dobijamo tačan link našeg zahteva koji je upućen serveru;
- code – dobijamo statusni kod našeg zahteva kao odgovor od servera;
- info() – dobijamo informacije o zaglavlju;
- read() – iščitavamo odgovor servera koji je tipa string; vrednost ovog atributa možemo iskoristiti za snimanje sadržaja dobavljene HTML stranice na fajl na našem disku;
- readlines() – iščitavamo odgovor servera, ali liniju po liniju. Povratna vrednost je tipa list, gde je svaki element jedna linija tog odgovora. Važno je napomenuti da, kao i kod čitanja fajlova, ako upotrebimo read() na odgovoru jednom, pokazivač će nam biti na kraju tog odgovora. Ukoliko prilikom ponovnog čitanja tog odgovora, taj pokazivač ne postavimo na početak odgovora – dobićemo prazan string u slučaju read() metode i praznu listu u slučaju readlines() metode, što se na našem primeru i vidi. Prosleđivanje lokacije odakle želimo početi sa čitanjem odgovora se vrši preko argumenta tipa int koji prosleđujemo metodi read(), odnosno metodi readlines().

Sajt https://httpbin.org konfigurisan je tako da pozivanjem https://httpbin.org/get dobijamo odgovor koji čini, zapravo, naš zahtev sa svim parametrima koje smo prosledili. Kako naš zahtev sadrži samo zaglavlje – to naše zaglavlje, pretvoreno u povratni odgovor, smo i dobili.

Prosleđivanje parametara pomoću urllib.parse je nešto komplikovanije i zahteva prvobitno kodiranje tih parametara u serverski pogodnom formatu, te spajanje tog parametra sa linkom sajta ka kojem šaljemo GET zahtev.

Kodiranje parametara se vrši pomoću urllib.parse biblioteke, koju moramo prvo uvesti.

**Napomena**

Na internetu se može naći dosta rezultata o urllib2 biblioteci, koja je zapravo biblioteka za Python2.x. U Python trojci biblioteka sa takvim imenom ne postoji. Tačnije, urllib2 je podeljen na urllib.request, urllib.response, urllib.parse i urllib.error biblioteke. Takođe, postoji i urllib3 biblioteka koja je razvijena od strane zajednice i koja se instalira pomoću komande *pip*.

Za samo kodiranje se koristi urlencode() metoda urllib.parse biblioteke. Parametar koji se prosleđuje je tipa podatka rečnik – gde su ključevi imena promenljivih koje će server tražiti, a vrednosti tih promenljivih – vrednosti za koje želimo da pitamo server. Na primer, ako želimo poslati serveru zahtev da nam dopremi sve korisnike koji se zovu John, pored osnovnog URL-a ka kojem šaljemo zahtev dodaćemo i first_name=John, pa bi tako jedan takav URL mogao izgledati ovako: https://httpbin.org/get?first_name=John. Karakter (?) govori serveru da od tog trenutka u URL-u počinju parametri po kojima želimo da se zahtev izvrši. Ako želimo više parametara, odvojićemo ih znakom (&):

```
https://httpbin.org/get?first_name=John&last_name=Johnson
```

Kao što vidimo, ako unapred znamo koje argumente želimo da pošaljemo, ovakav link možemo kreirati i bez kodiranja.

Nakon što smo kodirali svoje parametre linijom:

```
parsed_data = urllib.parse.urlencode({'first_name':'John',
'last_name':'Johnson'})
```

možemo napraviti i GET zahtev tako što linku jednostavno pridružimo parsed_data promenljivu koja je tipa string dok vodimo računa da između osnovnog URL-a i vrednosti promenljive parsed_data imamo karakter ?:

**Primer slanja GET zahteva sa argumentima**

```
import urllib.request

import urllib.parse

parsed_data = urllib.parse.urlencode({'first_name':'John',
'last_name':'Johnson'})

response =
urllib.request.urlopen('https://httpbin.org/get?{}'.format(parsed_data))
```

I u ovom slučaju response promenljiva je tipa http.client.HTTPResponse, pa se tako metode i atributi iz prethodnog primera mogu primeniti i ovde.

**Biblioteka requests**

Slanje GET zahteva je veoma jednostavno kada se koristi ova biblioteka, a pre slanja je moramo uvesti naredbom import requests. Za slanje GET zahteva koristimo metodu get() ovog modula kojoj prosleđujemo link ka kome želimo da pošaljemo zahtev. Povratna vrednost get() metode je Request objekat. To je sve što moramo učiniti za slanje GET zahteva pomoću ove biblioteke:

```
import requests

r = requests.get('https://httpbin.org/get')

print("URL: ",r.url)

print("Headers: ",r.headers)

print("Status code: ", r.status_code)

print("Text: ", r.text)

print("OK: ", r.ok)
```

Gde će rezultat da bude:

URL: https://httpbin.org/get

Headers: {'Date': 'Mon, 29 Jun 2020 01:56:42 GMT', 'Content-Type': 'application/json', 'Content-Length': '307', 'Connection': 'keep-alive', 'Server': 'gunicorn/19.9.0', 'Access-Control-Allow-Origin':

'*', 'Access-Control-Allow-Credentials': 'true'}
Status code: 200
Text: {
"args": {},
"headers": {
"Accept": "*/*",
"Accept-Encoding": "gzip, deflate",
"Host": "httpbin.org",
"User-Agent": "python-requests/2.21.0",
"X-Amzn-Trace-Id": "Root=1-5ef94a5a-3cc7cd380419f946d60cd559"
},

"origin": "24.135.243.74",

"url": "https://httpbin.org/get"
}
OK: True

Kao i urllib.request, requests biblioteka nam pruža određene metode i atribute kada je reč o Response objektu. Neke od njih smo upotrebili na našem primeru i to su:

- url atribut – ispisuje link našeg zahteva upućenog serveru, tip string;
- headers atribut – dobijamo informacije o zaglavlju, tip dict;
- status_code atribut – ispisuje statusni kod našeg zahteva, tip int;
- text atribut – ispisuje tekst serverskog odgovora, tip string. Kao što je slučaj i kod zahteva načinjenog urllib.request metodom, pošto nismo prosledili nijedan argument, odgovor je samo zaglavlje našeg zahteva prebačeno u odgovor. Takođe, pored atributa text postoji i atribut content, koji vraća bytes objekat. Vrednost ovog atributa možemo iskoristiti za snimanje sadržaja dobavljene HTML stranice na fajl na našem disku.
- ok atribut – veoma koristan atribut koji je tipa bool, a ispisuje True ili False u zavisnosti od toga da li je statusni kod ispod ili iznad 400. Ako je statusni kod 4xx ili 5xx, znači da je reč o klijentskoj, odnosno serverskoj grešci.

Za slanje argumenata uz naš GET zahtev, potrebno je malo izmeniti prethodni requests.get() poziv. Kao što je slučaj i kod urllib.request načina slanja GET zahteva sa argumentima, i ovde te argumente prosleđujemo kao tip rečnik, s tim što nije potrebno dodatno kodiranje, jer će to umesto nas obaviti sama biblioteka. Argumente prosleđujemo funkciji get() koristeći arugment params:

```
requests.get('https://httpbin.org/get',params={'first_name':'John',
'last_name':'Johnson'})
```

**Pitanje**

Kada koristimo requests biblioteku, koju njenu metodu koristimo za slanje GET zahteva ka serveru?

- make_get()
- urlopen()
- **get()**

**Objašnjenje:**

*Tačan odgovor je da se za slanje GET zahteva pri korišćenju requests biblioteke koristi metoda get(), kojoj u slučaju GET zahteva prosleđujemo sam link odakle zahtevamo podatke.*

## POST zahtev ka HTTP serveru

### Biblioteka urllib.request

Iz prethodnog primera smo videli da je za slanje GET zahteva pomoću urllib.request biblioteke potreban poziv urlopen() funkcije. Međutim, za slanje POST zahteva se koristi jedan korak više. Takođe, uz POST metodu uvek šaljemo i neke argumente, što nije slučaj kod GET metode, sa kojom tražimo podatke od servera. Na našem test sajtu, krajnja tačka zadužena za POST zahtev se nalazi na adresi: https://httpbin.org/post.

Pošto moramo poslati i argumente, sve prethodne instrukcije o enkodiranju koristeći urllib.parse se primenjuju i ovde. Nakon što kreiramo kodirani objekat koji sadrži naše argumente, prosleđujemo ga metodi Request() koja vraća objekat tipa Request. Ova metoda je deo urllib.request biblioteke i sa njom kreiramo zahtev tipa POST, ali ga još uvek ne šaljemo. Njoj prosleđujemo link ka kojem šaljemo zahtev kao i rezultat metode Request(), tačnije objekat tipa Request:

```
import urllib.request

import urllib.parse


parsed_data = urllib.parse.urlencode({'first_name':'John', 'last_name':'Johnson'})

req = urllib.request.Request('https://httpbin.org/post',bytes(parsed_data, encoding
= 'utf8'))

resp = urllib.request.urlopen(req)

print(resp.read())
```

Gde će rezultat da bude:

b'{\n "args": {}, \n "data": "", \n "files": {}, \n "form": {\n "first_name": "John", \n "last_name": "Johnson"\n }, \n "headers": {\n "Accept-Encoding": "identity", \n "Content-Length": "33", \n "Content-Type": "application/x-www-form-urlencoded", \n "Host": "httpbin.org", \n "User-Agent": "Python-urllib/3.7", \n "X-Amzn-Trace-Id": "Root=1-5ef95370-13173b845319291e1fa65ac7"\n }, \n "json": null, \n "origin": "24.135.243.74", \n "url": "https://httpbin.org/post"\n}\n'

Ako detaljnije pogledamo odgovor servera, videćemo svoje zaglavlje i argumente koje smo poslali. Takođe, atributi i metode koje smo koristili na http.client.HTTPResponse objektu prilikom slanja GET zahteva se mogu koristiti i u ovom slučaju.

**Biblioteka requests**

Slanje POST zahteva pomoću ove biblioteke je dosta slično slanju GET zahteva sa jednom malom razlikom – umesto get() metode koristimo post() metodu čijem argumentu params prosleđujemo tip rečnik:

```
r = requests.post('https://httpbin.org/post', params={'q':'wordpress',
'type':'title'})
```

Atributi koje smo primenjivali na Request objekat prilikom slanja GET zahteva važe i ovde.

**Napomena**

U oba slučaja je važno napomenuti da je potrebno da krajnja tačka, tačnije link ka kojem šaljemo zahtev, podržava primanje tog zahteva (npr. ako server na linku www.example.com/get ne podržava (ne očekuje) GET zahtev – dobićemo grešku).

## Skidanje fajlova sa mreže

Takođe, u radu sa HTTP protokolom kroz skripte ćemo često dolaziti u situaciju da nam je potrebno da skinemo određeni fajl – sliku, dokument, video itd. Obe ove biblioteke podržavaju ovu funkcionalnost.

### Skidanje fajla pomoću urllib.request biblioteke

Skidanje fajla uz pomoć ove biblioteke se vrši jednostavnim pozivom urlretrieve() metode kojoj prosleđujemo dva argumenta: kranji link na kojem se fajl nalazi i ime fajla (ekstenzija nije neophodna, ali je poželjna) u koji želimo smestiti skinuti fajl:

```
urllib.request.urlretrieve('https://www.python.org/static/img/python-
logo@2x.png', 'python_logo.png')
```

Na ovaj način smo sačuvali sliku u fajl po imenu python_logo.png koji se nalazi u direktorijumu gde i naš skript koji pokreće urlretrieve() metodu. Takođe je moguće zadati čitavu putanju gde želimo smestiti fajl:

```
urllib.request.urlretrieve('https://www.python.org/static/img/python-
logo@2x.png', 'C:\\Users\\user\\Documents\\python_logo.png')
```

Unutar drugog parametra metode urlretrieve, ukoliko navodimo lokalnu adresu gde želimo da sačuvamo fajl, potrebno je pisati dve kose crte ulevo umesto jedne, jer prva kosa crta crta može predstavljati escape karakter.

### Skidanje fajlova pomoću requests biblioteke

Skidanje fajlova pomoću ove biblioteke je jednostavno kao i pomoću urllib.request biblioteke. Sve što nam je potrebno jeste da pozovemo get() metodu kojoj zadajemo link do fajla. Nakon toga moramo ručno otvoriti fajl ugrađenom funkcijom open() i u njega smestiti podatke:

```
r = requests.get('https://www.python.org/static/img/python-logo@2x.png')

with open('python_logo.png', 'wb') as f:

    f.write(r.content)
```

Kod ovog pristupa je bitno napomenuti da se fajl mora otvoriti u binarnom modu namenjenom za pisanje – wb, i da umesto r.text koristimo r.content koji nam vraća bytes objekat.

**Napomena**

Kod oba pristupa, ako na lokaciji gde želimo da snimimo fajl na svom disku postoji fajl sa istim takvim imenom – biće zamenjen novim.

## Rezime

- Dve najpopularnije Python biblioteke za slanje zahteva ka HTTP serveru su urllib.request i requests.
- Za slanje GET zahteva, urllib.request biblioteka koristi metodu urlopen() kojoj prosleđujemo link servera odakle želimo da dobavimo podatke, dok biblioteka requests koristi metodu get(), kojoj se takođe prosleđuje link.
- Ako metodom GET želimo da pošaljemo i određene argumente uz link, ako koristimo biblioteku urllib.request, najpre moramo od tih argumenata napraviti rečnik, kodirati ga koristeći urllib.parse.urlencode() naredbu i dalje takav objekat proslediti kao dodatni argument urlopen() metode. Kada je reč o requests biblioteci, takođe argumente pretvaramo u rečnik, ali ih ne kodiramo, već ih zajedno sa linkom prosleđujemo kao argumente funkciji get().
- Za slanje POST zahteva, urllib.request biblioteka prvo ponovo kodira argumente na isti način kao što je slučaj kod GET zahteva, a zatim koristi novu metodu Request() kako bi kreirala objekat tipa zahtev koji se dalje zajedno sa linkom prosleđuje metodi urlopen(). Kada je reč o requests biblioteci, POST zahtev šaljemo metodom post(), kojoj prosleđujemo link, kao i podatak tipa rečnik koji sadrži argumente koje želimo da pošaljemo ka serveru.
- Skidanje fajlova sa mreže se, kada se koristi urllib.request biblioteka, vrši metodom urlretrieve(), kojoj prosleđujemo link do fajla i ime u koje želimo da ga sačuvamo, a kada je reč o requests biblioteci – to se vrši get() metodom, s tim što sami moramo izvršiti upis u fajl.